Your Name: Manu Bhargava Reddy Nannuri

Your Andrew ID: mnannuri

Homework 2

Statement of Assurance

I certify that all of the material that I submitted is my original work and that it is done only by me.

1 Experiment 1: Baselines

	Ranked Boolean	BM25 BOW	Indri BOW
P@10	0.1500	0.2900	0.2400
P@20	0.1800	0.3050	0.2750
P@30	0.1667	0.3267	0.2967
MAP	0.0566	0.1325	0.1275
Time	00:16	00:17	00:19

2 Experiment 2: Queries with Synonyms and Phrases

2.1 Queries

10:#NEAR/2(cheap internet)

12:djs

26:lower #NEAR/2(heart rate)

29:#SYN(#NEAR/1(ps 2) PS2 PLAYSTATION) games

33:#NEAR/1(elliptical trainer)

52:avp.url

71:living india

102:#NEAR/3(fickle creek farm)

149:uplift at #NEAR/5(yellowstone national park)

190:brooks brothers #SYN(clearance sale)

2.2 Query descriptions

10:#NEAR/2(cheap internet)

This is a common phrase and therefore a near operator is best suited to find the occurrence of this phrase.

12:djs

djs is a very common word and since there might be many web pages which happen to have djs listed on them. So the best matched models which take into account the document length and other factors might do a better job in weeding out irrelevant pages.

26:lower #NEAR/2(heart rate)

Heart rate is a popular phrase and hence applying the near operator would fetch documents which are relevant to heartrate only.

29:#SYN(#NEAR/1(ps 2) PS2 PLAYSTATION) games

ps 2 PS2 and playstation are interchangeably used in a wide variety of settings. Hence the syn operator would capture all those occurrences and fetch us those results which have games occuring along with the ps2 argument.

33:#NEAR/1(elliptical trainer)

Elliptical trainer is a commonly occuring phrase and hence searching for it using the near operator would fetch us the maximum number of results.

52:avp.url

avp looks like an acronym. Hence there is a higher probability that it might be present in a url or inlink text. Hence we formed the query using only avp.url

71:living india

India is a very specific term and hence just using it as single word woudl get better result. Using a near operator would narrow down the document list and hence might not rank the best documents the highest.

102:#NEAR/3(fickle creek farm)

Fickle creek farm is a name and hence these tokens always occur together. Hence the usage of the near operator might be apt to retrieve the relevant document.

149:uplift at #NEAR/5(yellowstone national park)

This information need talks about upliftment at Yellowstone National Park. A relevant result might have all the key words and the entire phrase "yellowstone national park". This is the rationale behind structuring the query this way.

190:brooks brothers #SYN(clearance sale)

Clearly brooks brother is a very specific word and we neeed not use any special operators for it. Using a syn operator on the clearance and sale term would retrieve more documents which are relevant.

2.3 Experimental Results

	Ranked Boolean	BM25 BOW	Indri BOW	Ranked Boolean Syn/Phr	BM25 Syn/Phr	Indri Syn/Phr
P@10	0.1500	0.2900	0.2400	0.3000	0.3100	0.2300
P@20	0.1800	0.3050	0.2750	0.2900	0.3250	0.2450
P@30	0.1667	0.3267	0.2967	0.2433	0.3300	0.2767
MAP	0.0566	0.1325	0.1275	0.0924	0.1520	0.1543
Time	00:16	00:17	0:19	00:15	00:17	00:18

2.4 Discussion

Ranked Boolean:

After using structured queries, the MAP precision numbers have improved. This is because of the additional constraints we put on the query terms with near and syn. The documents which are irrelevant might actually score higher in the absence of additional query operators because one term which is there in the document might be occurring many times in the document. This is in accordance with our beliefs.

BM25:

I expected the bm25 to perform way better when we put in more constraints as it had already performed better in the absence of any other query operators. But as per my experiments, I obtained slightly increased precision numbers.

One reason for this might be that, when using near operator we are performing an "**and**" operation and reducing the effective term frequency. This would reduce the score of the documents leading to aberrations in the results. This mainly happens because the term frequency is an important part of our score calculations.

Also, BM25 involves a summation over all its query terms. When we do a near operation on some of the query terms we are effectively reducing the number of the terms in the summation terms. This might reduce the score of those documents which might not have the required terms in a specified window size of the near operator.

Indri:

One reason which could explain the little improvement in precision numbers is that I have not used many syn queries. Using the syn queries would boost the PMLE(q|C). leading to higher scores for documents and also it would result in less number of products(of terms less than 1) as syn results give combined inverted lists.

The indri retrieval model performed similarly before and after using structured queries. I think it might be the same reason as to why the bm25 didn't perform well(writen in the previous paragraph).

Comparison between the retrieval models.

Running time: Obviously, the boolean ranked model is the quickest once because of its simple and minimal formula. It doesn't do the extensive computations like the indri and the bm25 retrieval models and hence runs very quickly. The okapi and bm25 models on the other hand do similar number of computations and hence have close running times.

Precision numbers:

For the bag of words queries, the best match retrieval models outperform the boolean ranked retrieval model. This is because boolean ranked model is an exact match model. Where as the bm25 and the indri models are best match models which account for the absence of some of the terms in the documents by using smoothing functions and relevant document scoring metrics. The document ranking.

3 Experiment 3: BM25 Parameter Adjustment

3.1 k_1

	k_1							

	1.2	0.8	1.1	1.4	2.0	3.0	5.0	100
P@10	0.2900	0.2400	0.2400	0.2400	0.2900	0.3000	0.3100	0.2100
P@20	0.3050	0.3000	0.3000	0.3000	0.3000	0.2950	0.3000	0.2050
P@30	0.3267	0.3267	0.3267	0.3233	0.3200	0.3133	0.3067	0.2300
MAP	0.1325	0.1255	0.1254	0.1252	0.1332	0.1334	0.1328	0.1048
Time	00:17	00:17	00:17	00:16	00:16	00:16	00:17	00:16

.

3.2 b

	b					
	0.75	0.05	0.25	0.45	0.75	0.95
P@10	0.2900	0.2700	0.2800	0.2600	0.2900	0.2500
P@20	0.3050	0.2850	0.3150	0.3000	0.3050	0.3100
P@30	0.3267	0.3000	0.3000	0.3133	0.3267	0.3333
MAP	0.1325	0.1219	0.1319	0.1324	0.1325	0.1291
Time	00:17	00:16	00:17	00:16	00:17	00:16

- the time are in the range of 16500ms to 17500 ms.

3.3 Discussion

As we vary k1 keeping b constant we see that the map values rise and then start dipping like a concave function. Initially i tried sampling the function only at values around the k1=1.2. The MAP value did not vary much, but as I conducted the experiments with increasing values of k1, the MAP score increased and then started to decrease somewhere between k1=3.0 and k1=5.0. Since Lucene and other report the optimal value of k1 to be 1.2 after testing on huge indexes and varied query set, I think the value of optimal k1 depends on the query set and this should be optimized according to the query set and the index. At values of k1 close to 0, we expect the precision numbers to be close to the binary models because we are completely ignoring the term frequency part of the formula by setting k1=0.

I observed similar results by varying the values of b between 0 and 1. The result tend to rise as we increase the value of b, then they attain a peak at a certain and value of b and then they tend to dip again. When b=0, we are totally ignoring the document length normalization part of the formula and this usually tends to prefer large documents. This means that we have a bias towards large documents and they tend to be ranked higher. So the actual relevant document might get ranked lower. When b=1, there is maximum document length normalization and we are over penalizing the large documents. This could make the actual relevant documents with large length get lower rank. The optimum value for this test set is somewhere between 0.75 and 0.9 and at that value we obtain optimum results.

4 Indri Parameter Adjustment

4.1 μ

	μ					
	2500	10	1600	2400	3200	10000
P@10	0.2400	0.2700	0.2300	0.2400	0.2100	0.2000
P@20	0.2750	0.3100	0.2850	0.2850	0.2700	0.29
P@30	0.2967	0.3133	0.3033	0.3000	0.2933	0.29
MAP	0.1275	0.0602	0.1311	0.1283	0.1247	0.0391
Time	00:19	00:19	00:19	00:19	00:19	00:19

4.2 λ

	λ					
	0.4	0.1	0.3	0.5	0.7	0.9
P@10	0.2400	0.1700	0.2100	0.2500	0.2700	0.2900
P@20	0.2750	0.2300	0.2750	0.2900	0.3100	0.3100
P@30	0.2967	0.2667	0.2800	0.2967	0.3000	0.3100
MAP	0.1275	0.1155	0.1231	0.1305	0.1346	0.1370
Time	00:19	00:19	00:19	00:19	00:19	00:19

4.3 Discussion

. I sampled the values of μ at the values of 10, 1600, 2400, 3200,10000. I specifically chose these values to observe how the precision numbers vary with different values of μ. I have observed that the MAP numbers are gradually increasing and have obtained a peak around the optimal value of 2500 and have started to decrease again. At extremely low values of μ, we lose the smoothing gains given to us by μ. We might lose out documents which don't have one of the query terms in it. This might result in a loss of precision as such missed out documents might be relevant. At extremely large values of μ, it might be the case that we are overshadowing the term frequency components and introducing unwanted noise in the formula calculation.

Since λ takes between values between o and 1, I conducted experiments at values of λ which span its entire length between 0 and 1. When λ is 0, we observe that the precision is low when compared to the higher values of λ. This is mainly because, when λ=0, the term frequency information is nullified and the queries are only ranked by the maximum likelihood estimates of the query terms given the corpus. This destroys the ranking order because all the documents have similar scores. This leads to low precision. As we go towards higher values of λ, our MAP numbers are increasing. This is because of the increasing weightage of the term frequency and the document relevance measure(idf). As we set our optimal value of λ at 0.4, we are supposed to get better results at this value. But my observations don't suggest this. I think this could be because of the choice of our query set. Due to the choice of our queries

terms, we are witnessing very little contribution from the mle estimates of the query terms given the corpus(the reference model). The second term only gains importance when a document is relevant despite the absence of a query term. The highly ranked document all have the query terms present in those documents.. I think due to the choice of our query terms, we might not be realising the significance of the second term(the term corresponding to the reference model).